

**2021年春季学期  
计算学部《软件构造》课程**

**Lab 1实验报告**

|  |  |
| --- | --- |
| 姓名 | 胡尼的孜 |
| 学号 | 1190201322 |
| 班号 | 1903012 |
| 电子邮件 | 1533460922@qq.com |
| 手机号码 | 18063113723 |

**目录**

[1 实验目标概述 1](#_Toc71131534)

[2 实验环境配置 1](#_Toc71131535)

[3 实验过程 1](#_Toc71131536)

[3.1 Magic Squares 1](#_Toc71131537)

[3.1.1 isLegalMagicSquare() 1](#_Toc71131538)

[3.1.2 generateMagicSquare() 1](#_Toc71131539)

[3.2 Turtle Graphics 1](#_Toc71131540)

[3.2.1 Problem 1: Clone and import 2](#_Toc71131541)

[3.2.2 Problem 3: Turtle graphics and drawSquare 2](#_Toc71131542)

[3.2.3 Problem 5: Drawing polygons 2](#_Toc71131543)

[3.2.4 Problem 6: Calculating Bearings 2](#_Toc71131544)

[3.2.5 Problem 7: Convex Hulls 2](#_Toc71131545)

[3.2.6 Problem 8: Personal art 2](#_Toc71131546)

[3.2.7 Submitting 2](#_Toc71131547)

[3.3 Social Network 2](#_Toc71131548)

[3.3.1 设计/实现FriendshipGraph类 2](#_Toc71131549)

[3.3.2 设计/实现Person类 2](#_Toc71131550)

[3.3.3 设计/实现客户端代码main() 2](#_Toc71131551)

[3.3.4 设计/实现测试用例 3](#_Toc71131552)

[4 实验进度记录 3](#_Toc71131553)

[5 实验过程中遇到的困难与解决途径 3](#_Toc71131554)

[6 实验过程中收获的经验、教训、感想 3](#_Toc71131555)

[6.1 实验过程中收获的经验和教训 3](#_Toc71131556)

[6.2 针对以下方面的感受 3](#_Toc71131557)

# 实验目标概述

本次实验通过求解三个问题，训练基本 Java 编程技能，能够利用 Java OO 开

发基本的功能模块，能够阅读理解已有代码框架并根据功能需求补全代码，能够

为所开发的代码编写基本的测试程序并完成测试，初步保证所开发代码的正确性。

另一方面，利用 Git 作为代码配置管理的工具，学会 Git 的基本使用方法。

⚫ 基本的 Java OO 编程

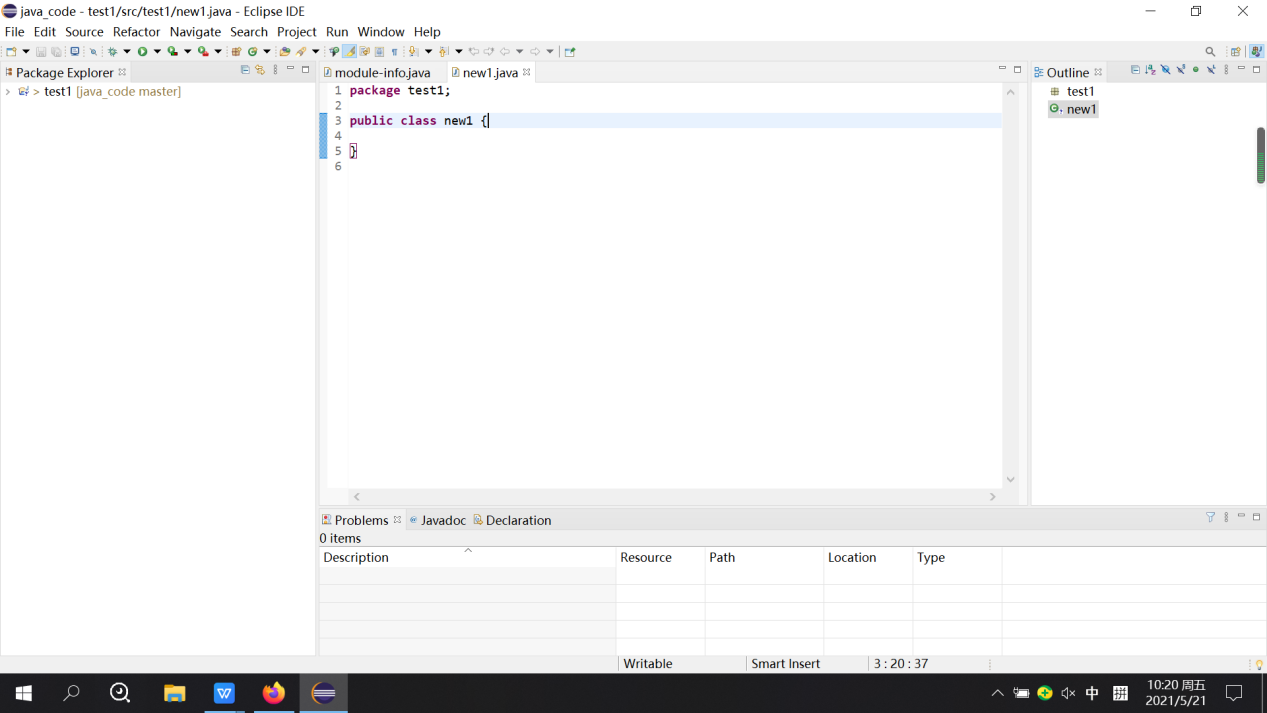
⚫ 基于 Eclipse/IDEA IDE 进行 Java 编程

⚫ 基于 JUnit 的测试

⚫ 基于 Git 的代码配置管理

# 实验环境配置

在官网下载了Eclipse和JDK并安装， 在GitHub网站下载安装了Gitbush，分为GUI（图形）和CMD（命令行）两种模式，并在网站上学习了相关教程，试着学习并使用了两种模式的使用。以下为Eclipse的使用界面截图。



在下载安装JDK的过程中出现了一些问题，最后发现是因为文件名出错，最后也解决了。在环境配置等方面在CSDN博客上找到了相关文章，解决了该问题。

以下是GitHub Lab1仓库的URL地址。

https://github.com/1190201322/HIT-Lab1-1190201322.git

# 实验过程

请仔细对照实验手册，针对四个问题中的每一项任务，在下面各节中记录你的实验过程、阐述你的设计思路和问题求解思路，可辅之以示意图或关键源代码加以说明（但无需把你的源代码全部粘贴过来！）。

为了条理清晰，可根据需要在各节增加三级标题。

## Magic Squares

A magic square of order *n* is an arrangement of n×n numbers, usually distinct integers, in a square, such that the n numbers in all rows, all columns, and both diagonals sum to the same constant .

根据题意，n阶幻方是n×n数在方格中的排列，通常是独立的，使得所有行中的n个数、所有列的n个数以及两者对角线都等于同一个常数。

根据题意，首先输入文件给出的n阶矩阵，判断该矩阵是否合法，若合法，判断是否是幻方。其次，输入一个正整数，如果是奇数，生成一个n阶矩阵，使其满足幻方性质，并保存到输出按文件中；如果是偶数，或者不是正整数，提示输入非法。然后判断生成的矩阵是否是幻方。

### isLegalMagicSquare()

该函数要判断一个矩阵是否为幻方。

首先读入文件：创建FileReader、BufferReader、StringBuilder对象，初始化line。然后逐行将字符串转换为整型矩阵存储，将非空readline的字符串分割

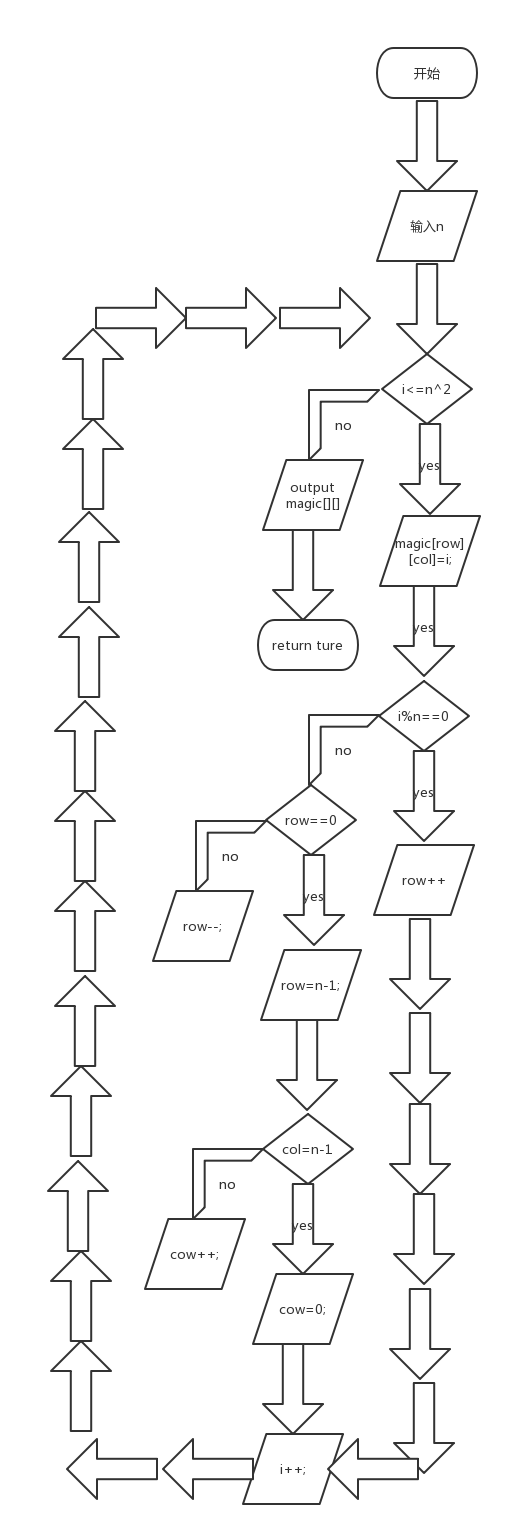
并去除头尾空格后转换为数字存储到二维数组中，再存储时判断该数字是否出现过，如果出现过就报错，否则用Boolean表标记。再然后，判断行列长度是否相等，计算两条斜线的和并比较，分别得出主对角线和次对角线的和，比较后若相等则记录，作为基准值；若不相等则报错。再计算每条纵线和横线和和并比较。（分别计算第i条横线和纵线的和，与基准值比较，若不相等则报错）。最后，确定是否为幻方。

### generateMagicSquare()

该函数要实现生成一个边长为奇数的幻方。

首先进行初始化，生成空矩阵。然后循环n\*n次填充矩阵。最后打开文件，打印结果。

具体过程如下：把1放在第一行正中，按以下规律排列剩下的(n×n－1)个数：每一个数放在前一个数的右上一格，如果这个数所要放的格行数小于1，则放在底行，仍然要放在上一个放置的数的右一列；如果这个数所要放的格列数大于n，则放在第1列，仍然要放在上一个放置的数的上一行；如果这个数所要放的格行数小于1，且列数大于n，那么就把它放在第n行第1列；如果这个数所要放的格已经有数填入，那么就把它放在上一个放置的数的下一行同一列。

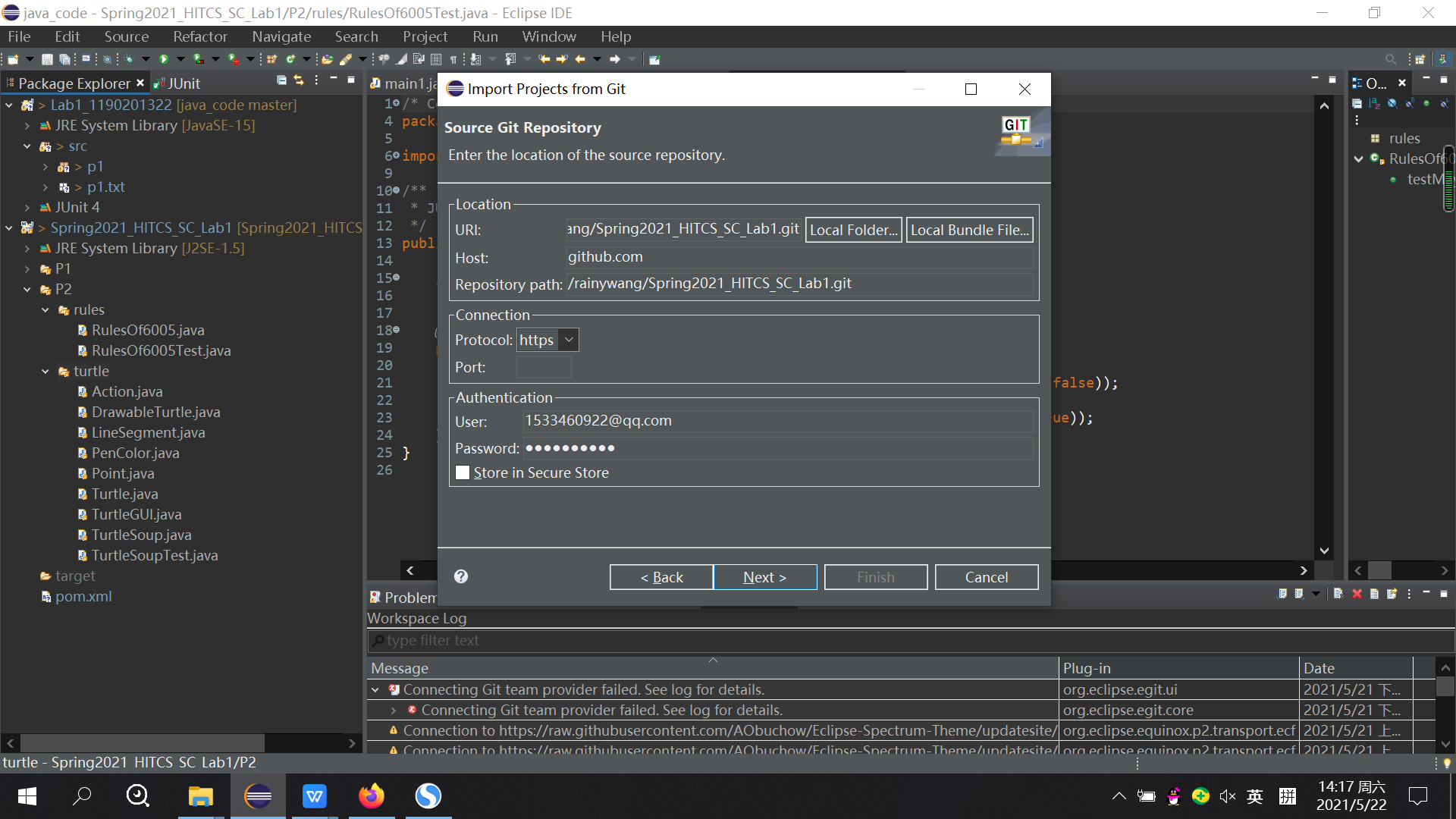


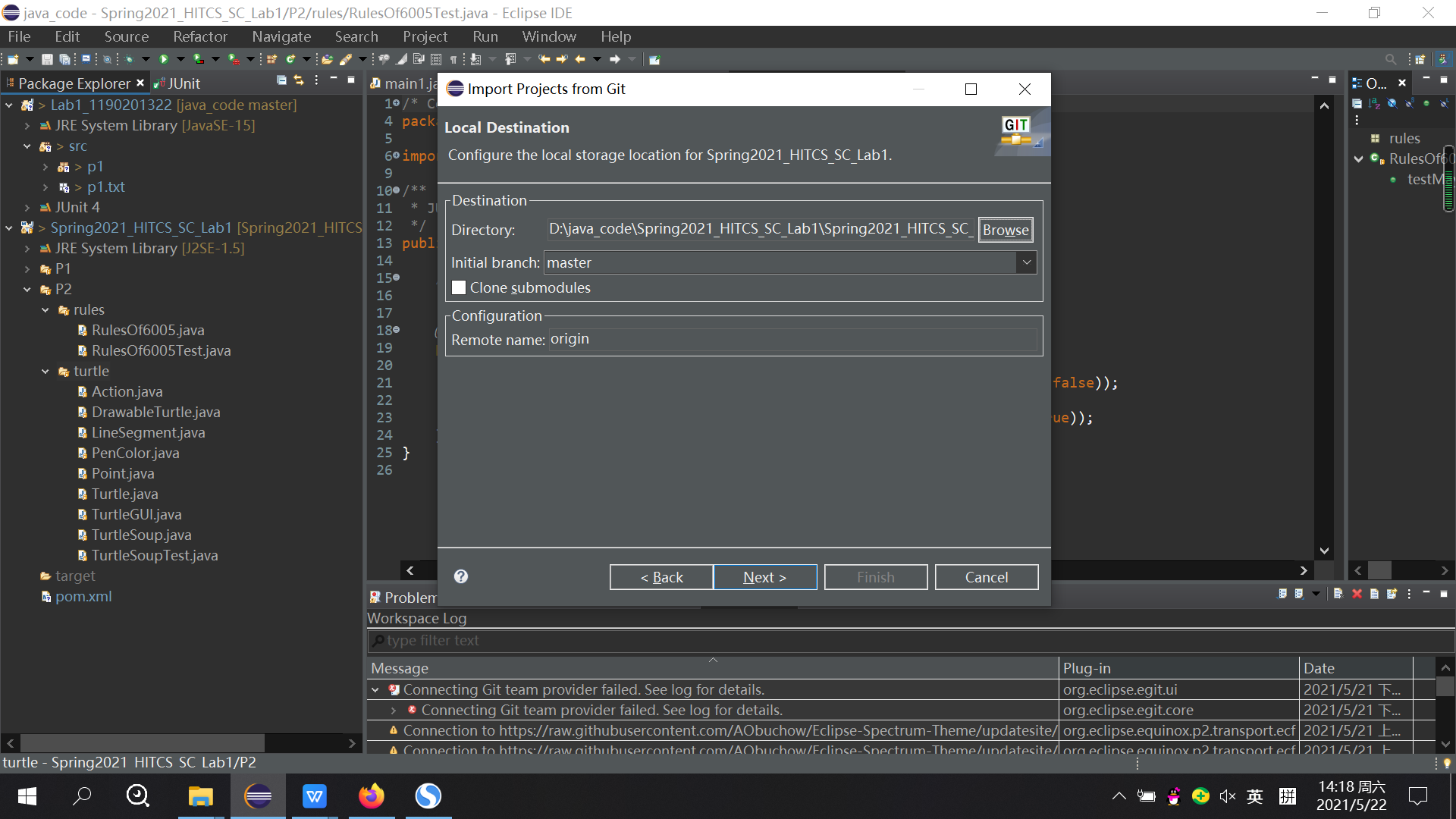
## Turtle Graphics

该任务需要我们clone已有的程序后，利用turtle按照要求画图，其中需要利用几何知识设计一些函数简化编程，最后可以发挥想象力进行Personal Art。首先分析turtle的package组成，了解类成员。

### Problem 1: Clone and import

1. 如何从GitHub获取该任务的代码：在进行第一项任务时利用Eclipse的import功能，把老师给的代码从Github中获取成功，在这次任务中继续使用即可。





2.在本地创建git仓库：通过使用Git Bash命令窗口，根据视频教程在本地创建了git仓库，命名为1190201322。

3.使用git管理本地开发：根据教程，可以通过命令行来管理开发。

### Problem 3: Turtle graphics and drawSquare

问题：给定爬行长度sideLength，使little turtle画一个边长为sideLength的正方形。

解决思路：画正方形，相当于先向前平移sideLength个单位，然后转90°，重复四遍即可。

代码如下：

public static void drawSquare(Turtle turtle, int sideLength) {

turtle.color(*PenColor*.***BLACK***);

for (int i = 0; i < 4; i++) {

turtle.forward(sideLength);

turtle.turn(90);

}

### Problem 5: Drawing polygons

(1)public static double calculateRegularPolygonAngle(int sides)

问题：给出正整数sides（>2），返回正sides边形的内角度数。

public static double calculateRegularPolygonAngle(int sides) {

return (double) 180.0 - (double) 360.0 / sides;

}

解决办法：用公式给出正整数sides（>2），返回正sides边形的内角度数。

（2 )public static int calculatePolygonSidesFromAngle(double angle)

问题：给出浮点数angle（0<angel<180），返回以angle为内角度数的正n边形边数。

解决办法：多边形外角和=360°，所以正n边形边数公式为：sides=360/(180-angle)。

1. public static void drawRegularPolygon(Turtle turtle,int sides,int sideLength)

问题：给出边数、边长，画一个正多边形。

解决办法：调用3.2.3.（2）的函数，求出正多边形的内角度数，然后用画正方形的方法画正多边形，只不过循环的次数改为sides次，turn的角度改为算出的角度。

private JPanel contentPane;

public static void drawRegularPolygon(Turtle turtle, int sides, int sideLength) {

turtle.color(*PenColor*.***BLACK***);

for (int i = 0; i < sides; i++) {

turtle.forward(sideLength);

turtle.turn((double) 180.0 - *calculateRegularPolygonAngle*(sides));}

}

### Problem 6: Calculating Bearings

1. Public static double calculateBearingToPoint(double currentBearing,int currentX,int currentY,int targetX,int targetY)

**问题：**计算当前向量与目标向量的夹角（当前向量转过多少度后达到目标向量）。其中当前向量由当前点、当前方向决定，目标方向由当前点、目标点决定，当前方向由当前方向与竖直向上方向（y轴正方向）的夹角表示，目标向量的方向由当前点与目标点的连线与竖直向上方向（y轴正方向）的夹角表示。

**解决办法：**目标向量的求法如下：先求当前点与目标点的连线与y轴夹角的正切值，tan=(targetX-currentX)/(targetY-currentY)，然后调用Math库中的atan将其转化为弧度，再由公式——角度=弧度\*180/PI——求得角度值，即为当前向量与y轴正方向的夹角，与currentBearing（当前方向）作差，得解。

（2）public static List<Double> calculateBearings(List<Integer> xCoords, List<Integer> yCoords) {

**问题：**以列表形式给出一系列点的横纵坐标。假设海龟从给出的第一点开始，面朝上(即0度)。对于其后的每一点，假设海龟在移至前一点时仍朝其所面对的方向前进。

**解决办法：**对相邻两个点调用3.2.4.（1）的函数求夹角，保存在列表中，返回该列表。

### Problem 7: Convex Hulls

**问题：**凸包问题，给出一组点的坐标，求最少的点的集合，使其他所有点都在这些点围成的闭合凸多边形内。

**解决办法：**使用gift-wrapping算法。

以任意凸包上的点建立一个极角坐标系，该点连结其它所有点的极角中，该点逆时针方向的第一凸包点到该点极角最小。首先找到最左边的点，这个点必然在凸包上，然后计算该点连接点极角最小的，这里计算有技巧，算法中进行toright测试,直到找到到最右端的点，找到P1后，就可以从P1开始，接着顺次找到P2，又以P2为起点一直进行到最后一个点为止。

### Problem 8: Personal art

# 任务：自己画一种图形，详细的见Github。

# 思路；利用JAVA画图turtle彩螺旋线。

### Submitting

在安装好Git，注册好Github账号并且创建仓库的基础上，利用Git bash来初始化Git,再添加远程仓库的URL，（命令行为：git remote add origin https://github.com/.../x...x.git）。然后再用命令行git pull添加上传文件，并且添加修改日志，用命令行：git commit -m "Initial commit"。最后用命令行git push -u origin master进行PUSH操作，然后刷新Github网页便能看见上传文件。其他具体操作可通过视频教程或者CSDN博客等方式进行自学。

## Social Network

设计一张社交网络图，基于连接人与人，并且能计算任意两人之间的联系情况。网络图基于两个类，分别是FriendshipGraph类和Person类。

### 设计/实现FriendshipGraph类

**存储结构：**Set<Person> persons = **new** HashSet<Person>();

用HashSet存放Graph里的节点（人）。

1. public void addVertex(Person person);

作用：将person这个节点添加到HashSet中，相当于在图中插入一个独立的节点。

PS：如果有一个节点的name属性与待插入节点的name属性相同，就抛出重名提示，不插入这个节点。

1. public void addEdge(Person p1,Person p2);

作用：在p1到p2节点之间连一条单向边。用邻接表的方式存储边，

Person类的add\_friend方法，在p1的friends集合里加入p2这个节点。

（为了避免重边，插入边之前先扫描p1的friends集合，判断是否有一个集合内的点，其name属性与p2相同，如果有，证明p1与p2之间已经存在这样的单向边，抛出重边提示，不插入这条边。）

1. public int getDistance(Person p1,Person p2);

作用：对给定的两个节点，求两点间的最短路。

实现过程：用BFS实现最短路。

1. 初始化图中的每个点的vis属性为false；
2. 新建一个队列，用来存放已搜索过的节点。
3. p1作为根节点，p1.vis置为true，p1加入队列；
4. 搜索p1的friends集合中未访问过的节点（vis属性为false），如果有一个满足要求的子节点等于p2，即找到一条最短路，结束搜索，返回当前路径长度distance；否则，将满足要求的子节点依次加入队列，修改其vis属性值为true；
5. 所有满足条件的子节点扫描完之后，当前路径++；
6. 取出队首节点，重复第3、4、5步操作，直到队列为空，或找到最短路径。
7. 值得注意的一点是存在p1和p2不相连的情况，这时BFS搜索到最后队列为空，扔找不到一条最短路径，因此需要在一开始设置一个标记变量flag，初始化为false，找到子节点为p2时修改flag的值为true，这样搜索结束时，如果flag的值没有发生变化，就说明两个节点不连通，返回值设为-1。

### 设计/实现Person类

1. public String Name;存放当前节点的名字；
2. HashSet<String> NameSet = new HashSet<>();：存放以当前节点为起点的所有边的终点；
3. private boolean vis;在BFS中标记当前节点是否被访问过；
4. public void addVertex(Person newPerson)向当前节点的friends集合中添加一个新节点，即在当前节点与目标节点之间添加一条单向边；

### 设计/实现客户端代码main()

调用FriendshipGraph类的正确方法：

FriendshipGraph graph = new FriendshipGraph();

得到的graph相当于一张完整的有向图，可调用内部方法来实现节点的插入、边的插入和两点间最短路径的查询。

### 设计/实现测试用例

设计思路：（1）测试新创建的图是否为空，初始化检测；

（2）插入新节点，测试addVertex方法是否能够实现功能；

（3）插入name属性相同的节点，测试addVertex方法是否按约定抛出提示；

（4）插入新的有效边，测试addEdge方法是否能够实现功能；

（5）插入重复边，测试addEdge方法是否按约定抛出提示；

（6）选取两个连通点，测试getDistance方法是否返回正确结果；

（7）选取两个非连通点，测试getDistance方法是否返回-1；

# 实验进度记录

请使用表格方式记录你的进度情况，以超过半小时的连续编程时间为一行。

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 时间段 | 任务 | 实际完成情况 |
| 2021-05-07 | 18:30-22:30 | 学习了JAVA语言的一些基础知识。 | 按计划完成 |
| 2021-05-10 | 18:30-21:00 | 下载并安装了JDK以及Eclipse等软件，并且学习了相关教程 | 按计划完成 |
| 2021-05-11 | 18:30-22:30 | 在实验课上下载安装了Git，并且注册和实现了创建本地库和远程库 | 遇到困难，在和同学讨论过程中完成了一大部分，剩余的在教程中学习了 |
| 2021-05-12 | 14:00-18:30 | 看了问题一，解决了一部分，过程很慢 | 完成了一半左右，遇到困难 |
| 2021-05-18 | 18:30-21:30 | 和同学讨论了问题一，并且看了一部分第二题 | 这次完成了一大半，过程很慢 |
| 2021-05-19 | 18:30-22:30 | 在实验课上和同学一起进行了第二题 | 按计划完成了第二题 |
| 2021-05-20 | 18:30-22:30 | 看了一大部分第三题 | 按计划完成 |
| 2021-05-21 | 18:30-22:30 | 完成了第三题 | 按计划完成 |
| 2021-05-22 | 18:30-22:30 | 撰写了实验报告，并提交 | 按计划完成 |
|  |  |  |  |

# 实验过程中遇到的困难与解决途径

|  |  |
| --- | --- |
| 遇到的困难 | 解决途径 |
| 把3.1中的老师给定的文件从Github导入到Eclipse时一直不能克隆成功。 | 在CSDN上找到了几篇文章，通过其帮助用指令  git config --global --unset http.proxy  解决了该问题。 |
| 在没有认真阅读实验要求之前就下载了JDK和Eclipse，由于两者不匹配，出现了问题 | 重新下载了JDK11，然后与Eclipse 调试 |
| 在进行第二题时，不会使用turtle gui，导致画图出现了很大的问题，最后问题也没有解决的很好 | 会在课下进行自主学习这方面的知识，尽量解决这些问题。 |
| 由于Github网站访问很艰难，在提交和下载文件等方面出现了很多问题 | 可以使用自己的热点；可以多试几次；可以等。 |

# 实验过程中收获的经验、教训、感想

## 实验过程中收获的经验和教训

这次实验让我清楚的知道了自己在哪些方面的不足，虽然也明白这次试验量很大，难度也不小，但是身边仍有人完成得很好，甚至非常优秀，而我连能看懂题目一开始都出现了问题。同时，由于时间没有安排好，前一周的时间由于其他科目考试未能真正开始做实验，导致后面的实验完成的都很匆忙，很多细节都不够好。下次会提前学习好java以及其他相关知识，尽量早点开始进行，并且也会努力在每一个细节做到最好。

## 针对以下方面的感受

1. Java编程语言是否对你的口味？

虽然不是很熟练，但是还感觉还不错，尤其是其中一些知识点和之前学过的知识能进行融会贯通。

1. 关于Eclipse IDE；

使用起来比较顺手，不需要等它编译，页面也比较好看，其他功能也不错。

1. 关于Git和GitHub；

这两者是我觉得比较难的地方，可能是刚刚开始使用不归熟练，导致每次使用命令行等时比较痛苦。

1. 关于CMU和MIT的作业；

首先英语是大问题，其次，难度比较大，对我来说，题型也比较新。

1. 关于本实验的工作量、难度、deadline；

工作量太大了，难度也不小，时间可能 和其他考试时间以及计算机系统实验时间撞到一起，体会了火烧眉毛的感觉。

1. 关于初接触“软件构造”课程；

首先，上课节奏很快，而且内容也很难，对我来说，老师ppt上面的很多知识点对我来说甚是陌生，让人害怕；其次，这门课和之前得专业课难度不是一个层次的，可能是更接近于工作时的难度和强度，但是往后还是会努力的去完成任务，慢慢进步的，希望老师下次能多给一些时间。